We are IntechOpen, the world’s leading publisher of Open Access books
Built by scientists, for scientists

4,200 Open access books available
116,000 International authors and editors
125M Downloads

154 Countries delivered to
TOP 1% Our authors are among the most cited scientists
12.2% Contributors from top 500 universities

WEB OF SCIENCE™
Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected. For more information visit www.intechopen.com
1. Introduction

Linear spectral transform techniques such as the discrete Fourier transform and wavelet analysis over real and complex fields have been routinely applied in the literature (Burrus et al. (1998); Strang & Nuygen (1996)). Furthermore, extensions of these techniques over finite fields (Blahut & Burrus (1991); Caire et al. (1993)) have led to applications in the areas of information theory and error control coding (Blahut (2003); Dodd (2003); Sakk (2002); Wicker (1994)). The goal of this chapter is to review the Galois Field Fourier Transform, the associated convolution theorem and its application in the field of error control coding. In doing so, an interesting connection will be established relating the convolution theorem over finite fields to error control codes designed using finite geometries (Blahut (2003); Lin & Costello (1983); Wicker (1994)).

While a complete exposition of the field of error control would be out of context for this chapter, we refer the interested reader to the recent characterizations of Low-Density Parity Check (LDPC) codes (Pusane et al. (2011); Smarandache et al. (2009); Xia & Fu (2008)). Such formulations have led to a resurgence of interest in the design (Kou et al. (2001); O.Vontobel et al. (2005); Tang et al. (2005); Vandendriesscher (2010)) and decoding (Kou et al. (2001); Li et al. (2010); Liu & Pados (2005); Ngatched et al. (2009); Tang et al. (2005); Zhang et al. (2010)) of finite geometry codes. The formulation in this chapter is meant to serve as a guiding principle relating finite geometric properties to algebraic ones. The vehicle we have chosen to demonstrate these relationships is an example from the field of error control. In particular, we show how a generalized Fourier-like convolution theorem can be applied as a decoding methodology for finite geometry codes.

We begin in Section 2 by reviewing the Galois Field Fourier Transform (GFFT) followed by an overview of error control coding in Section 3. In addition, in Section 3.1 it is demonstrated how the GFFT can be applied within the context of error control coding. Section 4 then goes on to generalize these results to linear transformations using Pascal’s triangle as an example. The combinatorics of such a transformation naturally lead to the design of codes derivable from...
finite geometries. Finally, Sections 5 and 6 conclude this chapter by deriving and applying the generalized convolution theorem.

2. The Galois Field Fourier Transform

We are particularly interested in the case of finite fields where \( p \) is a prime number and \( \alpha \in GF(p^n) \) is an element of order \( n \). The Galois Field Fourier Transform (GFFT) and its inverse of a vector \( v = \{ v_0, v_1, \ldots, v_{n-1} \} \) over \( GF(p) \) of length \( n \) can be related via the equations:

\[
V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i \quad j = 0, \ldots, n-1
\]

and

\[
v_j = (n)^{-1} \sum_{j=0}^{n-1} \alpha^{-ij} V_j \quad i = 0, \ldots, n-1.
\]

For any vector \( f \) over \( GF(p) \) where the above equations hold true, we define

\[
\mathcal{F}(v) \equiv V = \{ V_0, V_1, \ldots, V_{n-1} \}
\]

as the GFFT of \( v \) and

\[
\mathcal{F}^{-1}(V) = v = \{ v_0, v_1, \ldots, v_{n-1} \}
\]

as the inverse GFFT of \( F \).

Using this formulation, given two vectors

\[
v = \{ v_0, v_1, \ldots, v_{n-1} \}
\]

\[
w = \{ w_0, w_1, \ldots, w_{n-1} \}
\]

over \( GF(p) \) and their associated transforms

\[
\mathcal{F}(v) = V = \{ V_0, V_1, \ldots, V_{n-1} \}
\]

\[
\mathcal{F}(w) = W = \{ W_0, W_1, \ldots, W_{n-1} \},
\]

the familiar convolution theorem can be demonstrated to hold true for the finite field case. Specifically, computing

\[
x_j = \sum_{k=0}^{n-1} v_k w_{j-k}
\]

is equivalent to computing

\[
x_j = \mathcal{F}^{-1}(V_jW_j).
\]

3. Error control coding

Given a message encoded as a vector \( \mu \) of length \( k \) over \( GF(p) \), the goal of error control coding (ECC) is to transform the message vector into a code vector \( C \) of length \( n > k \) in a way that causes \( C \) to be robust to errors arising over a communication channel (such as a wireless
link, fiber optic cable, etc). Rather than the message vector $\mu$, it is the code vector $C$ that is transmitted over a channel where the receiver is only able to observe a received vector $\hat{C}$. Ideally, in the absence of any noise, it should be the case that $\hat{C} = C$. On the other hand, if noise is present on the channel, the method used to transform (i.e. 'encode') the message $\mu$ into the code vector $C$ provides a way to recover $\mu$ from $\hat{C}$. The basic strategy behind ECC is, given a message,

a. Embed a $k$ dimensional message vector $\mu$ in a larger vector space of dimension $n$ to create the code vector $C$.

b. The addition of channel noise converts $C$ into the received vector $\hat{C}$.

c. If the channel noise does not cause $\hat{C}$ to be confused with other possible encodings, the original code vector $C$ can be recovered using some predetermined decoding scheme. Conceptually speaking, the $\hat{C}$ that lies within a predefined noise 'sphere' with respect to the original $C$ will be decoded as the (ideally) unique $C$; hence, $\mu$ can be recovered as well. The size of the noise sphere (which is designed as part of the code) determines how many errors can be corrected.

The general idea behind ECC then is to find a $C$ that minimizes $||C - \hat{C}||$; however, numerically determining the minimum distance solution is wrought with dimensionality issues that can lead to computational intractability. Hence, classes of codes have been devised that relate the message encoding method to the decoding algorithm. Such algorithms are often iterative (Blahut (2003); Lin & Costello (1983); Wicker & Kim (2003)) and converge upon the optimal solution by exploiting the mathematical structure designed into the code.

Two important quantities in the field of ECC are the Hamming weight and the Hamming distance. Consider two vectors $v$ and $w$ of length $n$ over $GF(p)$.

**Definition 3.1.** The Hamming weight $w_H(v)$ of a vector $v$ is defined as the number of non-zero components in $v$.

**Definition 3.2.** The Hamming distance between $v$ and $w$ is defined as the number of components that differ between $v$ and $w$.

For example, over $GF(3)$, assuming $n = 5$, $v = \{0 2 1 0 2\}$ and $w = \{0 2 2 1 2\}$, according to the above definitions we have that $w_H(v) = 3$, $w_H(w) = 4$ and $d_H(v, w) = 2$.

An important quantity for defining the noise sphere is referred to as $d_{min}$ which is the minimum Hamming distance between all code vectors defined in the code class. To correct up to $t$ errors in any code vector, it turns out that $d_{min} = 2t + 1$. Furthermore, when the ECC is a linear code, a major simplification arises where $d_{min}$ is simply the minimum Hamming weight computed over all non-zero code vectors in the code class.

### 3.1 Application of the GFFT to Reed-Solomon codes

The GFFT and the convolution theorem have been applied in the field of error control coding for the construction of a class of linear codes known as Reed-Solomon codes (Blahut (2003); Wicker (1994)). The algorithm for encoding a message vector $\mu$ over $GF(p^m)$ of length $k$ is
quite straightforward. To be able to correct up to $t$ errors, create a vector of length $n$ by appending $\mu$ with $2t$ consecutive zeros. The code vector $C$ is then derived by computing the inverse GFFT of the appended construction. One approach to proving that this construction is capable of correcting up to $t$ errors involves applying the GFFT convolution theorem. Specifically, given a code vector $C$, a locator vector $\Lambda$ must be defined such that $C_j \Lambda_j = 0$ for all $j = 0, \cdots, n$. Letting $c$ and $\lambda$ denote the GFFT of $C$ and $\Lambda$, the convolution theorem implies $c \ast \lambda = 0$. Based upon the convolution approach, the conclusion can be reached that the inverse GFFT construction leads to Reed-Solomon codes capable of correcting up to $t$ errors in the code vector (Blahut (2003); Wicker (1994)).

The key feature of the GFFT approach to constructing Reed-Solomon codes described above is that restrictions are placed on the position and the number of zeros appended to the message vector. To summarize:

i. Addition of zeros to the message vector $\mu$ of length $k$ is performed at prescribed locations.
ii. The resulting vector is then inverse transformed in order to compute the code vector $C$.
iii. The error correcting properties of this code can be demonstrated by applying the convolution theorem.

In this work, one of our goals is to demonstrate that, given other linear transformations inducing a convolution theorem, the above steps can be generalized to other classes of codes. As we shall see, the key is to define the transform and the structure of how zeros are introduced into the message vector.

4. Pascal codes

4.1 The Pascal matrix over finite fields

Let us now focus our attention on the case of $GF(p)$ where $p$ is prime. Our starting point will be:

Definition 4.1. Let $p$ be a prime number, then the $ij^{th}$ entry of a $p^m \times p^m$ $m^{th}$ order Pascal matrix $P_{p^m}$ over $GF(p)$ is defined as

$$p_{ij} = \binom{j}{i}(j-i)!^{-1} \mod p$$

$$= \binom{j}{i} \mod p$$

(7)

for $i, j = 0, 1, \ldots, p^m - 1$ and, by convention, if $i > j$, then $p_{ij} = 0$.

In other words, $P_{p^m}$ is an upper triangular matrix whose non-zero entries are the elements of Pascal’s triangle taken mod $p$. For the purposes of this work, it is useful to observe that $P_{p^m}$ also has a Kronecker product description (Sakk & Wicker (2003)):

$$P_{p^m} = P_p \otimes P_{p^{m-1}} \mod p$$

(8)

where $P_p$ is a $1^{st}$ order Pascal matrix.

www.intechopen.com
Example 4.2. Consider the binary case where $p = 2$ and $m = 3$. Equation (8) gives

$$P_{2^3} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}.$$ 

Example 4.3. Consider the ternary case where $p = 3$ and $m = 2$. Equation (8) gives

$$P_{3^2} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 2 & 0 & 1 & 2 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 2 \\
0 & 0 & 0 & 1 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}.$$ 

4.2 The inverse of the Pascal matrix

As Section 5 will require understanding $Q_{p^m} \equiv P_{p^m}^{-1}$, we introduce the following.

Observation 4.4. Let $p$ be prime and let $Q_p$ be the $p \times p$ matrix defined by

$$q_{ij} = \begin{cases}
(-1)^{i-j} \binom{i}{j} & \text{mod } p \\
0 & \text{otherwise}
\end{cases} \text{ for } i,j=0,1,\ldots,p-1. \quad (9)$$

Then $Q_p = P_p^{-1} \pmod{p}$.

This result easily follows from the integer case (Call & Velleman (1993); Heller (1963)). Furthermore, it has been demonstrated that (Sakk (2002)).

Observation 4.5. If $p$ is prime and $P_p$ is a 1st order Pascal matrix over $GF(p)$, then

$$P_p^p \pmod{p} = I_p \quad (10)$$

where $I_p = p \times p$ identity matrix.

Hence, it easily follows that

Corollary 4.6. If $p$ is prime and $P_p$ is a Pascal matrix over $GF(p)$, then

$$Q_p = P_p^{-1} \pmod{p} = P_p^{p-1} \pmod{p}. \quad (11)$$
Example 4.7. A Pascal matrix over $GF(5)$ and its inverse:

$$
P_p = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
0 & 1 & 2 & 3 & 4 \\
0 & 0 & 1 & 3 & 1 \\
0 & 0 & 0 & 1 & 4 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}, \quad Q_p = P_p^4 = \begin{bmatrix}
1 & 4 & 1 & 4 & 1 \\
0 & 1 & 3 & 3 & 1 \\
0 & 0 & 1 & 2 & 1 \\
0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}.
$$

Based upon Equation (8), it should be clear that

$$
Q_p^m = Q_p \otimes Q_{p^{m-1}} \mod p.
$$

(12)

Finally, based upon Equation (10), it also follows that, for the $m^{th}$ order case,

$$
P_p^m \mod p = I_p^m
$$

where $I_p^m = p^m \times p^m$ identity matrix. In a manner similar to the $m = 1$ case, this characterization provides a path to computing the $m^{th}$ order inverse

$$
Q_p^m \equiv P_p^{-1} \mod p = P_p^{p^m-1} \mod p.
$$

(14)

4.3 Error control codes designed from Pascal matrices

In a manner similar to the GFFT approach to Reed-Solomon codes summarized in Section 3.1, it has been pointed out that $P_p^m$ can also be used to transform message vectors with the appropriate coordinates set equal to zero (Sakk & Wicker (2003)). More precisely, we have the following:

Definition 4.8. Consider an $m^{th}$ order Pascal matrix over $GF(p)$ and let $r$ be an integer such that $0 \leq r \leq m(p-1)$. Also, consider the $p$-ary expansion of an index

$$
i = i_0 p^0 + i_1 p^1 + \cdots + i_{m-1} p^{m-1}
$$

where $0 \leq i_j \leq p-1$ for $0 \leq j \leq m-1$. A codeword $c$ for an $r^{th}$ order Pascal code of length $p^m$, denoted by $P_p(r,m)$, is generated by

$$
C = \mu P_p^m
$$

(15)

where

$$
\mu = \left( \mu_0 \mu_1 \cdots \mu_{p^m-1} \right)
$$

is a message vector of length $p^m - 1$ such that $\mu_i \in GF(p)$,

$$
\begin{cases}
\mu_i = 0 & \text{if } w_p(i) > r \\
\mu_i \neq 0 & \text{if } w_p(i) \leq r
\end{cases}
$$

(16)
The Fourier Convolution Theorem over Finite Fields: Extensions of its Application to Error Control Coding

and

\[ w_p(i) = \sum_{j=0}^{m-1} i_j. \]

Error control codes derived from the \(m\)th order Pascal matrix over GF(2) (i.e. binary data) have been related (Forney (1988); Massey et al. (1973)) to a class of codes known as \(r\)th order binary Reed-Muller codes \(RM(r, m)\) of length \(2^m\) (MacWilliams & Sloane (1977); Wicker (1994)). In addition, it has been further demonstrated (Sakk (2002)) that \(P_2(r, m)\) codes over GF(2) are equivalent to \(RM(r, m)\) codes with minimum distance \(d_{\text{min}} = 2^{m-r}\). These observations have been extended where it has been demonstrated that \(P_p(r, m)\) codes over GF(p) are equivalent to generalized Reed-Muller codes (GRM) codes (Sakk (2002)).

To place this class of codes in the same context as that outlined in Section 3.1, we must show how to introduce zeros into the message vector, apply the Pascal matrix as the linear transformation and, based upon this transformation, introduce a convolution theorem. From the definition above, a given code is specified by choosing \(p, m\) and a value of \(0 \leq r \leq m(p - 1)\). The code vector length then becomes \(n = p^m\); and, for this class of codes, a given value of \(r\) defines the length \(k\) of the message. The rest of the \(n - k\) components of \(\mu\) must be set to zero in a systematic way that leads to the minimum distance property of the code.

**Example 4.9.** Consider \(P_{2^2}\) in Example 4.2 (hence, \(n = 2^3 = 8\)) and a message vector \(\mu = (\mu_0, \mu_1, \ldots, \mu_7)\) and let \(s\) be the number of consecutive zeros in the vector \(\mu\) for a given value of \(r\):

- \(r = 0 (d_{\text{min}} = 8): s = 7 \quad \mu = (\mu_0, 0, 0, 0, 0, 0, 0, 0) \quad (k = 1)\)
- \(r = 1 (d_{\text{min}} = 4): s = 3 \quad \mu = (\mu_0, \mu_1, \mu_2, 0, \mu_4, 0, 0, 0) \quad (k = 4)\)
- \(r = 2 (d_{\text{min}} = 2): s = 1 \quad \mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, 0) \quad (k = 7)\)
- \(r = 3 (d_{\text{min}} = 1): s = 0 \quad \mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7) \quad (k = 8)\)

**Example 4.10.** Consider \(P_{3^2}\) in Example 4.3 (hence, \(n = 3^3 = 27\)) and a message vector \(\mu = (\mu_0, \mu_1, \ldots, \mu_8)\) and let \(s\) be the number of consecutive zeros in the vector \(\mu\) for a given value of \(r\):

- \(r = 0 (d_{\text{min}} = 9): s = 8 \quad \mu = (\mu_0, 0, 0, 0, 0, 0, 0, 0, 0) \quad (k = 1)\)
- \(r = 1 (d_{\text{min}} = 6): s = 5 \quad \mu = (\mu_0, \mu_1, 0, \mu_3, 0, 0, 0, 0, 0) \quad (k = 3)\)
- \(r = 2 (d_{\text{min}} = 3): s = 2 \quad \mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, 0, \mu_6, 0, 0) \quad (k = 6)\)
- \(r = 3 (d_{\text{min}} = 2): s = 1 \quad \mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, 0) \quad (k = 8)\)
- \(r = 4 (d_{\text{min}} = 1): s = 0 \quad \mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8) \quad (k = 9)\)

In the above examples, \(d_{\text{min}}\) is shown in parentheses for each value of \(r\); furthermore, observe that \(d_{\text{min}} = s + 1\). Recalling for a moment the GFFT approach to Reed-Solomon code design, the minimum distance of a code where the message vector has \(n - k\) consecutive zeros can be shown to be \(d_{\text{min}} = n - k + 1\) (Blahut (2003); Wicker (1994)). It is apparent that, by using a Pascal matrix as the transform, a result similar to that of the GFFT can be ascertained. The major difference is that, for Reed-Solomon codes, the string of zeros must occur at the end of the message vector before applying the GFFT to create \(C\). For \(P(r, m)\), in addition to the string of consecutive zeros, based upon the structure of \(P_{p^m}\), zeros must also be dispersed in other positions within \(\mu\) to form code vectors \(C = \mu P_{p^m}\).
5. Extensions of the Fourier convolution theorem over finite fields

The convolution operation involves relating the componentwise product of two vectors in one domain to the convolution of their transforms (Blahut & Burrus (1991)). Many linear transforms have well-defined convolution operations. For instance, the Hadamard transform yields the so-called logical or ‘dyadic’ convolution operation (Ahmed et al. (1973); Dodd (2003); Robinson (1972)). In this chapter, we develop extensions of the convolution theorem that can be used to reveal useful properties of other classes of codes. As an example, we demonstrate how the GFFT approach can be applied to describe generalized Reed-Muller codes (Blahut (2003)).

To begin the formulation, we consider the componentwise product $\gamma_j = \mu_j \lambda_j$ of two vectors $\mu = (\mu_0 ... \mu_{n-1})$ and $\lambda = (\lambda_0 ... \lambda_{n-1})$. Furthermore, we consider matrix transforms such that $C \equiv \mu P_{\mu}$ and $\Lambda \equiv \lambda P_{\mu}$ or, equivalently, $\mu = C Q_{\mu}$ and $\lambda = \Lambda Q_{\mu}$ where $(P_{\mu})^{-1} \equiv Q_{\mu}$. (here, ‘$\mu$’ denotes the message vector and ‘$C$’ denotes the code vector). We demonstrate a formulation analogous to the convolution operation that describes $\gamma = \Gamma Q_{\mu}$:

$\Gamma_j = \sum_{j=0}^{n-1} \eta_j p_{ji} \mod p \quad i = 0, 1, ..., n - 1$

$= \sum_{j=0}^{n-1} (\mu_j \lambda_j) p_{ji} \mod p$

$= \sum_{j=0}^{n-1} \mu_j \left( \sum_{k=0}^{n-1} \Lambda_k q_{kj} p_{ji} \right) \mod p \quad (17)$

$= \sum_{k=0}^{n-1} \Lambda_k \left( \sum_{j=0}^{n-1} \mu_j q_{kj} p_{ji} \right) \mod p$

$= \sum_{k=0}^{n-1} \Lambda_k T_{i,k} \mod p \quad i = 0, 1, ..., n - 1$

where $n = p^m$.

Notice that if we are dealing with familiar spectral transforms such as the Fourier or the Hadamard transform (where $P$ denotes the forward transform and $Q$ denotes the inverse transform), $T_{i,k}$ takes on a simple form. This is because the product $q_{kj} p_{ji}$ in $\sum_{j=0}^{n-1} \mu_j q_{kj} p_{ji}$ reduces to a term that enables us to take the transform of $\mu$ as $C f(i,k) = \frac{1}{n} \left( \sum_{j=0}^{n-1} \mu_j P_{j,f(i,k)} \right)$. For the case of the Fourier transform $f(i,k) = i - k$ and $T_{i,k} = C(i-k)$; as expected, one ends up with the convolution theorem (Blahut (2003); Wicker (1994)). In the case of a Hadamard transform, $f(i,k) = i \oplus k$ (where $\oplus$ denotes bit-by-bit addition of the binary expansions of $i$ and $k$) and $T_{i,k} = C(i\oplus k)$. Here, the bit-by-bit addition $\oplus$ of the binary expansions of $i$ and $k$ over GF(2) would result in the dyadic convolution (Ahmed et al. (1973); Robinson (1972)).

For the codes in this presentation, the $q_{kj} p_{ji}$ term in the above summation leads to a convolution theorem that depends on the matrix $P_{\mu}$. Furthermore, this theorem can also be applied to demonstrate how to decode $C$ to recover the message vector $\mu$. In Equation (17)
\( q_{kj} = (-1)^{j-k} \mod p \) and \( p_{ji} = \binom{j}{i} \mod p \); therefore, the product \( q_{kj}p_{ji} \) will not lead to an expression that readily reduces the inner summation to a single term. To see why, let’s write out \( T_{i,k} \) as follows:

\[
T_i = (T_{i,0}, T_{i,1}, ..., T_{i,n-1})
\]

\[
\equiv (\mu_0 \mu_1 ..., \mu_{n-1}) \begin{bmatrix}
q_{00}p_{00} & q_{01}p_{01} & \cdots & q_{0(n-1)0}p_{00} \\
q_{01}p_{01} & q_{11}p_{01} & \cdots & q_{0(n-1)1}p_{11} \\
\vdots & \vdots & \ddots & \vdots \\
q_{0(n-1)0}p_{(n-1)0} & q_{1(n-1)0}p_{(n-1)1} & \cdots & q_{0(n-1)(n-1)}p_{(n-1)(n-1)}
\end{bmatrix}
\]

\[
= (\mu_0 \mu_1 ..., \mu_{n-1}) \begin{bmatrix}
p_{00} & q_{01} & \cdots & q_{0(n-1)0} \\
p_{01} & q_{11} & \cdots & q_{0(n-1)1} \\
\vdots & \vdots & \ddots & \vdots \\
p_{(n-1)0} & q_{1(n-1)0} & \cdots & q_{0(n-1)(n-1)}
\end{bmatrix}
\]

\[
\equiv \mu D_i Q_{\mu}^T
\]

where \( T \) denotes the matrix transpose.

**Observation 5.1.** The components of the vector \( T_i = (T_{i,0}, T_{i,1}, ..., T_{i,n-1}) \) can be written as a linear combination of the components of \( C = (C_0, ..., C_{n-1}) \).

**Proof:**

Let

\[
M_i \equiv D_i Q_{\mu}^T
\]

where \( D_i \) is defined in Equation (18) and

\[
A_i \equiv Q_{\mu}M_i \equiv Q_{\mu}D_i Q_{\mu}^T
\]

\[
\Rightarrow M_i = P_{\mu}A_i.
\]

Then,

\[
T_i = \mu M_i = \mu P_{\mu}A_i = CA_i.
\]

Combining this result with Equation (17) we conclude

\[
\Gamma_i = \sum_{k=0}^{n-1} \Lambda_k T_{i,k} \mod p \quad i = 0, 1, ..., n-1
\]

\[
= \sum_{k=0}^{n-1} \Lambda_k (CA_i)_k \mod p \quad i = 0, 1, ..., n-1
\]

So, instead of \( T_{i,k} \) reducing to one single component of the vector \( C \) (as one might expect from a typical convolution operation), the Pascal convolution requires a linear combination of the components of \( C \). Although this operation is slightly more complicated than the Fourier approach, the identity in Equation (8) does induce a simplification.

**Observation 5.2.** (Symbolic Computation of Pascal Convolution)

For the 1st order case where \( n = p \) and \( i = 0, ..., p - 1 \), using Equation (19) let \( \tilde{M}_i \equiv M_i, \)

www.intechopen.com
using Equation (18) let $\hat{D}_i \equiv D_i$ and let $\hat{A}_i \equiv Q_p \hat{M}_i$. Then, for any $0 \leq j \leq p^m - 1$ where $j = j_0 p^0 + j_1 p^1 + \ldots + j_{m-1} p^{m-1}$ and $A_j \equiv Q_p^m M_j$,

$$A_j = \hat{A}_{j_{m-1}} \otimes \ldots \otimes \hat{A}_{j_1} \otimes \hat{A}_{j_0}$$

(23)

where $M_j \equiv D_j Q_p^m$.

Proof: The statement is clearly true for the first order case $m = 1$ since $j = j_0$. By induction let $j = j_0 p^0 + j_1 p^1 + \ldots + j_{m-1} p^{m-1}$ and assume that

$$D_j = \hat{D}_{j_{m-1}} \otimes \ldots \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0}$$

where $0 \leq j_k \leq p - 1$ for all $k = 0, \ldots, m - 1$. Consider any $j' = j_0 p^0 + j_1 p^1 + \ldots + j_{m-1} p^{m-1} + j_m p^m$ and apply Equation (18) along with Lucas’ theorem to obtain the following intermediate result:

$$D_{j_n} \otimes \hat{D}_{j_{n-1}} \otimes \ldots \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0} = D_{j_n} \otimes D_j$$

$$= \begin{bmatrix}
\binom{j_0}{1}
& \ldots & \binom{j_n}{1}
& \ldots & \binom{j_{p-1}}{1}
& \ldots & \binom{j_{m-1}}{1}
& \ldots & \binom{j_{m-1}}{p-1}
\end{bmatrix}
\otimes
\begin{bmatrix}
\binom{j_0}{1}
& \ldots & \binom{j_1}{1}
& \ldots & \binom{j_{p-1}}{1}
& \ldots & \binom{j_{m-1}}{1}
& \ldots & \binom{j_{m-1}}{p-1}
\end{bmatrix}
$$

(24)

Therefore, $D_j = \hat{D}_{j_{m-1}} \otimes \ldots \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0}$ is true. Next, successively apply the identity $(AC) \otimes (BD) = (A \otimes B)(C \otimes D)$ to obtain:

$$\hat{M}_{j_{m-1}} \otimes \ldots \otimes \hat{M}_{j_1} \otimes \hat{M}_{j_0} = (\hat{D}_{j_{m-1}} Q_p^1) \otimes \ldots \otimes (\hat{D}_{j_1} Q_p^1) \otimes (\hat{D}_{j_0} Q_p^1)$$

$$= (\hat{D}_{j_{m-1}} \otimes \ldots \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0})(Q_p^1 \otimes Q_p^1 \otimes \ldots \otimes Q_p^1)$$

$$= D_j Q_p^m$$

$$= M_j$$

Finally, we arrive at the desired conclusion

$$(\hat{A}_{j_{m-1}} \otimes \ldots \otimes \hat{A}_{j_1} \otimes \hat{A}_{j_0}) = (Q_p \hat{M}_{j_{m-1}}) \otimes \ldots \otimes (Q_p \hat{M}_{j_1}) \otimes (Q_p \hat{M}_{j_0})$$

$$= (Q_p \otimes Q_p \otimes \ldots Q_p)(\hat{M}_{j_{m-1}} \otimes \ldots \otimes \hat{M}_{j_1} \otimes \hat{M}_{j_0})$$

$$= Q_p^m M_j$$

$$= A_j.$$
Observation 5.2 tells us that, in order to calculate $T_j = CA_j$ for arbitrary $n = p^m$, one need only calculate $\hat{A}_i$ for $i = 0, \ldots, p - 1$ and then take successive Kronecker products. The initial set of $\hat{A}_i$ for $i = 0, \ldots, p - 1$ can easily be calculated by referring back to Equation (20) where $\hat{A}_i = \hat{Q}_p \hat{M}_i = \hat{Q}_p \hat{D}_i \hat{Q}_p^T$.

An interesting property concerning the $A_j$ is that the sum

$$
\sum_{i=0}^{p^m-1} A_i = \sum_{i=0}^{p^m-1} \hat{Q}_p \hat{D}_i \hat{Q}_p^T
$$

(where the sum is taken mod $p$) is a matrix of ones. This follows from two observations. First, from the definition of $D_i$ in Equation (18), $\sum_{i=0}^{p^m-1} D_i$ is a matrix whose $(p^m - 1, p^m - 1)$ entry is one and all other entries are zero. Second, it can also be demonstrated that the last column of $Q_{p^m}$ must be a column of ones. Therefore, $\hat{Q}_p \sum_{i=0}^{p^m-1} \hat{D}_i \hat{Q}_p^T = \sum_{i=0}^{p^m-1} A_i$ is a matrix of ones.

**Example 5.3.** For $p = 2$, the 1st order case $n = p$ gives $i = 0, 1$; hence, over GF(2),

$$
P_p = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad Q_p = P_p = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix},
$$

we calculate

$$
\hat{A}_0 = \hat{Q}_p \hat{D}_0 \hat{Q}_p^T = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},
\quad \hat{A}_1 = \hat{Q}_p \hat{D}_1 \hat{Q}_p^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}.
$$

From Observation 5.2, to obtain the $A_j$ for $n = p^2$ and $j = 0, 1, 2, 3$, one need only take successive Kronecker products as:

$$
A_0 = \hat{A}_0 \otimes \hat{A}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_1 = \hat{A}_0 \otimes \hat{A}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},
\quad A_2 = \hat{A}_1 \otimes \hat{A}_0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_3 = \hat{A}_1 \otimes \hat{A}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.
$$

As expected, the $A_i$ are symmetric matrices. Also, notice, as mentioned above, that $\sum_{i=0}^{p^m-1} A_i$ is a matrix of ones. For the case where $n = p^2$, let us now apply Observation 5.1 to calculate the Pascal convolution of the vectors $C = (C_0, C_1, C_2, C_3)$ and $\Lambda = (A_0, A_1, A_2, A_3)$. Using Equation (22), we have:

$$
\Gamma_0 = \Lambda_0 C_0 + \Lambda_1(0) + \Lambda_2(0) + \Lambda_3(0),
\Gamma_1 = \Lambda_0 C_1 + \Lambda_1(C_0 + C_1) + \Lambda_2(0) + \Lambda_3(0),
\Gamma_2 = \Lambda_0 C_2 + \Lambda_1(0) + \Lambda_2(C_0 + C_2) + \Lambda_3(0),
\Gamma_3 = \Lambda_0 C_3 + \Lambda_1(C_2 + C_3) + \Lambda_2(C_1 + C_3) + \Lambda_3(C_0 + C_1 + C_2 + C_3).
$$

www.intechopen.com
To close this section, we draw some immediate conclusions from Equation (25):

- Because of the Kronecker product, a good deal of self-similar structure can be observed in the resulting vector \( \Gamma \). For instance, the coefficient of \( \Lambda_1 \) in \( \Gamma_1 \) can be computed by adding the coefficient of \( \Lambda_0 \) in \( \Gamma_0 \) to the coefficient of \( \Lambda_0 \) in \( \Gamma_1 \). The coefficients of \( \Lambda_2 \) and \( \Lambda_3 \) in \( \Gamma_2 \) and \( \Gamma_3 \) can be computed by adding the coefficients of \( \Lambda_0 \) and \( \Lambda_1 \) in \( \Gamma_0 \) and \( \Gamma_1 \) to the coefficients of \( \Lambda_0 \) and \( \Lambda_1 \) in \( \Gamma_2 \) and \( \Gamma_3 \), and so on.

- Looking at the result columnwise, the set of coefficients associated with a given \( \Lambda_i \) appear to be the checksums for an \( R(r, 2) \) binary Reed-Muller code ((MacWilliams & Sloane, 1977, p.385-388), (Wicker, 1994, p.155-165)). As pointed out in the next section, although this observation is true for the binary case, an orthogonal set of checksums for \( p > 2 \) will not come about by this method. It is the dual of the Pascal convolution that will lead to the decoding of GRM codes.

6. Majority logic decoding using Pascal convolution

GRM codes fall into a larger category of codes known as Euclidean geometry codes (Blahut (2003); Lin & Costello (1983); MacWilliams & Sloane (1977); Wicker (1994)) where it is well-known that a technique known as 'majority logic decoding' (MLD) can be used to recover the message vector. Based upon statements made in Section 4, it should be clear that Pascal codes are also MLD. However, the role played by the Pascal convolution in the decoding strategy is worthy of mention. As pointed out in the conclusions of Example 5.3, the checksums of a majority logic decoding (MLD) scheme for GRM codes can be derived using the dual of the convolution relation derived above. We now demonstrate this observation more clearly.

Because of the similar forms of \( P_{pm} \) and \( Q_{pm} \) the dual convolution relation is easily derived from the inverse transform. Consider the componentwise product \( \Gamma_j = C_j \Lambda_j \) of two vectors where \( C = \mu P_{pm} \) and \( \Lambda = \lambda P_{pm} \):

\[
\gamma_i = \sum_{j=0}^{n-1} \Gamma_j q_{ji} \mod p \quad i = 0, 1, ..., n-1
\]

\[
= \sum_{j=0}^{n-1} (C_j \Lambda_j) q_{ji} \mod p
\]

\[
= \sum_{j=0}^{n-1} C_j (\sum_{k=0}^{n-1} \lambda_k p_{kj} q_{ji}) \mod p
\]

\[
= \sum_{k=0}^{n-1} \lambda_k (\sum_{j=0}^{n-1} C_j p_{kj} q_{ji}) \mod p
\]

\[
= \sum_{k=0}^{n-1} \lambda_k s_{i,k} \mod p \quad i = 0, 1, ..., n-1
\]
where \( n = p^m \). Similar to Equation (18), one can also show that
\[
s_i = (s_{i,0} \ s_{i,1} \ ... \ s_{i,n-1})
\]
\[
= \mathcal{F} \Delta_i P_{p^n}^T
\]
which can also be written as
\[
s_i = \mu P_{p^n} \Delta_i P_{p^n}^T
\]
where \( \Delta_i \) is a diagonal matrix with elements \((q_{0i} \ q_{1i} \ ... \ q_{(n-1)i})\) along its diagonal. Furthermore, if we define
\[
B_j \equiv P_{p^n} \Delta_i P_{p^n}^T
\]
then results similar to Observations 5.1 and 5.2 can also be demonstrated. However, in proving the dual of Observation 5.2 there is one difference be aware of. Since \( q_{ji} = (-1)^{i-j} (\binom{i}{j}) \), the Kronecker product in the dual of Equation (24) will contain extra factors of \((-1)^{i-j}\). To achieve the equality \( \Delta_i = \hat{\Delta}_{i,n-1} \otimes \cdots \otimes \hat{\Delta}_{i,j} \otimes \hat{\Delta}_{j,0} \) where \( j = j_0 p^0 + j_1 p^1 + \cdots + j_{m-1} p^{m-1} \) the following identity will be required:
\[
(-1)^k = (-1)^{k_0 p^0 + k_1 p^1 + \cdots + k_{m-1} p^{m-1}}
\]
\[
= (-1)^{k_0}((-1)^p)^{k_1}((-1)^p)^{k_2} \cdots ((-1)^p)^{k_{m-1}}
\]
\[
= (-1)^{\sum_{i=0}^{m-1} k_i}
\]
for any \( 0 \leq k \leq p^m - 1 \) where we have applied \( a^{p^r} = a \) for any \( a \in GF(p) \). Then, following the proof of Observation 5.2, it is straightforward to show that for any \( 0 \leq j \leq p^m - 1 \) where \( j = j_0 p^0 + j_1 p^1 + \cdots + j_{m-1} p^{m-1} \),
\[
B_j = \hat{B}_{j,n-1} \otimes \cdots \otimes \hat{B}_{j,i} \otimes \hat{B}_{j,0}
\]
(30)
where
\[
\hat{B}_{j,i} = P_{p^n} \hat{\Delta}_{i,j} P_{p^n}^T
\]
In Section 4, we explained that the form of message vectors when applying \( P_{p^n} \) as the transformation where the message vector \( \mu = (j_0, \ ..., \ \mu_{p^m-1}) \) should have all components \( \mu_j = 0 \) if \( w_p(j) > r \) (see Examples 4.9 and 4.10). To see how this formulation can lead to a decoding scheme, let us examine the case where \( p = 2, m = 2 \) and \( r = 1 \) (i.e. a 1st order binary Reed-Muller code of length 4). Consider first using Equations (26) and (27) to calculate Pascal convolution of the vectors \( \mu = (\mu_0, \mu_1, \mu_2, \mu_3) \) and \( \lambda = (\lambda_0, \lambda_1, \lambda_2, \lambda_3) \):
\[
\begin{align*}
00 & : \quad \gamma_0 = \lambda_0 C_0 \quad + \quad \lambda_1 (0) \quad + \quad \lambda_2 (0) \quad + \quad \lambda_3 (0) \\
01 & : \quad \gamma_1 = \lambda_0 (C_0 + C_1) \quad + \quad \lambda_1 C_1 \quad + \quad \lambda_2 (0) \quad + \quad \lambda_3 (0) \\
10 & : \quad \gamma_2 = \lambda_0 (C_0 + C_2) \quad + \quad \lambda_1 (0) \quad + \quad \lambda_2 C_2 \quad + \quad \lambda_3 (0) \\
11 & : \quad \gamma_3 = \lambda_0 (\sum_{i=0}^{3} C_i) \quad + \quad \lambda_1 (C_1 + C_3) \quad + \quad \lambda_2 (C_2 + C_3) \quad + \quad \lambda_3 C_3
\end{align*}
\]
where the binary expansion of the $\gamma$ index has been explicitly written out at the beginning of each row. Next, consider Equations (26) and (28) to calculate the same convolution:

\[
\begin{align*}
00 & : \gamma_0 = \lambda_0 \mu_0 + \lambda_1(0) + \lambda_2(0) + \lambda_3(0) \\
01 & : \gamma_1 = \lambda_0 \mu_1 + \lambda_1(\mu_0 + \mu_1) + \lambda_2(0) + \lambda_3(0) \\
10 & : \gamma_2 = \lambda_0 \mu_2 + \lambda_1(0) + \lambda_2(\mu_0 + \mu_2) + \lambda_3(0) \\
11 & : \gamma_3 = \lambda_0 \mu_3 + \lambda_1(\mu_2 + \mu_3) + \lambda_2(\mu_1 + \mu_3) + \lambda_3(\sum_{i=0}^{3} \mu_i)
\end{align*}
\]

Since, for $P_2(1,2), \mu = (\mu_0, \mu_1, \mu_2, 0)$, this set of equations can be simplified as

\[
\begin{align*}
00 & : \gamma_0 = \lambda_0 \mu_0 + \lambda_1(0) + \lambda_2(0) + \lambda_3(0) \\
01 & : \gamma_1 = \lambda_0 \mu_1 + \lambda_1(\mu_0 + \mu_1) + \lambda_2(0) + \lambda_3(0) \\
10 & : \gamma_2 = \lambda_0 \mu_2 + \lambda_1(0) + \lambda_2(\mu_0 + \mu_2) + \lambda_3(0) \\
11 & : \gamma_3 = \lambda_0 \mu_3 + \lambda_1(\mu_2 + \mu_3) + \lambda_2(\mu_1 + \mu_3) + \lambda_3(\sum_{i=0}^{3} \mu_i)
\end{align*}
\]

Equations (31) and (32) must hold for any vector $\lambda$. Therefore, for a specific $\gamma_j$, we can equate the coefficients of the $\lambda_i$ in Equation (31) with those in Equation (32). So, for example, we end with the result that

\[
\begin{align*}
\mu_2 &= C_0 + C_2 \\
\mu_2 &= C_1 + C_3
\end{align*}
\]

and

\[
\begin{align*}
\mu_1 &= C_0 + C_1 \\
\mu_1 &= C_2 + C_3.
\end{align*}
\]

For this first order $r = 1$ code, we can generate a set of checksums using a simple algorithm. Start at an index $i$ of $\gamma$ such that $w_2(i) = 1$ and equate Equations (31) and (32) along a diagonal path in order to 'collect' all checksum equations associated with $\mu_i$. For example, the bold symbols in Equation (32) generate the checksums for $\mu_i$. It turns out that these diagonal equations actually generate what are known as the 'incidence vectors' of the MLD strategy (Blahut (2003); MacWilliams & Sloane (1977); Wicker (1994)).

We now provide an algorithm for $GF(p)$ to show how the Pascal convolution approach is equivalent to a typical MLD using finite Euclidean geometry (Wicker, 1994, p.155-165). The interesting aspect of this algorithm is that the Pascal convolution generates the correct checksums for any $GF(p)$. Consider a $P_p(r,m)$ code where $C = \mu P_p$ such that $\mu_j = 0$ if $w_p(j) > r$:

(0) Let $j = r$.

(1) Let $S_j$ be the set of indices $i$ such that $w_p(i) = j$.

(2) Apply Equation (27) to calculate $\gamma$.

(3) Apply Equation (28) to calculate $\gamma$ (these equations will simplify based upon which of the $\mu_i$ are zero).
(4) For each \(i \in S_j\), start at \(\lambda_0\) associated with \(\gamma_i\) and construct checksum equations by equating the result in Step (2) with that of Step (3) along a diagonal path (i.e. - starting at \(k=0\), choose the coefficient of \(\lambda_k\) associated with \(\gamma_{i+k}\)).

(5) For \(i \in S_j\), create estimates \(\hat{\mu}_i\) by a majority logic decision on the checksums.

(6) \(j = j - 1\). If \(j < 0\), stop.

(7) Remove the estimated components as:

\[
\hat{C} = \hat{\mu} P_m
\]

\[
\check{C} \equiv C - \hat{C} = (\mu - \hat{\mu}) P_m
\]

(8) Adjust \(\mu\) to reflect the change in step (7) as follows. Construct a new vector \(\tilde{\mu}\) where \(\tilde{\mu}_i = \mu_i\) if \(i \in S_j\) and \(\tilde{\mu}_i = 0\) otherwise. Then let

\[
\hat{\mu} \equiv \mu - \tilde{\mu}
\]

(9) Let \(C = \check{C}\) and \(\mu = \hat{\mu}\) and go to Step (1).

As with typical MLD schemes, this algorithm starts with the highest order \(r\) to obtain estimates of the code vector components and then successively estimates the lower order components.

**Example 6.1.** Let \(p = 3, m = 2\) and \(r = 2\). Consider decoding a \(P_3(2, 2)\) code. From Example 4.10,

\[
\mu = (\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, 0, 0, 0, 0)
\]

Also, we know that \(P_3(2, 2)\) has \(d_{\text{min}} = 3\) implying that we can correct a single error. Therefore, we expect that the MLD equations should have at least three checksums.

(0) Start with \(j = 2\).

(1) Let \(S_2 = \{2, 4, 6\}\) (i.e. - \(i = i_0 + i_1 p\) such that \(w_3(i) = 2\)).

(2,3,4) Rather than write out the equations for \(\gamma_i\), we summarize by equating the results of step (2) and step (3):

\[
i = 2:\quad \mu_2 = c_0 + c_1 + c_2
\]
\[
\mu_2 = c_3 + c_4 + c_5
\]
\[
\mu_2 = c_6 + c_7 + c_8
\]

\[
i = 4:\quad 2\mu_4 = 2c_1 + c_2 + c_4 + c_5
\]
\[
2\mu_4 = 2c_3 + c_4 + c_6 + 2c_5
\]
\[
\mu_4 = c_4 + 2c_5 + 2c_7 + c_8
\]
\[
\mu_6 = c_0 + c_3 + c_6
\]

\[
i = 6:\quad \mu_6 = c_1 + c_4 + c_7
\]
\[
\mu_6 = c_2 + c_5 + c_8
\]
After estimating the message components dictated by $S_2$ (step (5)), remove the code estimates from $C$ (step (7)) and begin work on $S_1$ where now (step(8)) $\mu_i = 0$ if $w_p(i) > 1$. For $S_1$, we have the checksums:

\begin{align*}
\mu_1 &= 2c_0 + c_1 \\
2\mu_1 &= c_1 + 2c_2 \\
\mu_1 &= 2c_3 + c_4 \\
2\mu_1 &= c_4 + 2c_5 \\
\mu_1 &= 2c_6 + c_7 \\
2\mu_1 &= c_7 + 2c_8 \\
\mu_3 &= 2c_0 + c_3 \\
\mu_3 &= 2c_1 + c_4 \\
\mu_3 &= 2c_2 + c_5 \\
2\mu_3 &= c_3 + 2c_6 \\
2\mu_3 &= c_4 + 2c_7 \\
2\mu_3 &= c_5 + 2c_8 \\
\end{align*}

$i = 1 :$

After estimating the message components dictated by $S_3$, once again, remove the code estimates from $C$ and begin work on $S_0$ where now $\mu_i = 0$ if $w_p(i) > 0$. At this stage, with all other components of $\mu = 0$ except $\mu_0$, we are left with $\mu = C$ (i.e. nine estimate of the check on $\mu_0$).

7. Conclusions

When considering the design of error control codes, it is interesting to look for guiding principles that can account for whole classes of codes. In this presentation, we have shown how the GFFT convolution approach to Reed-Solomon codes can be extended to other classes of codes such as generalized Reed-Muller codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Convolution Principle</th>
<th>Decoding Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reed-Solomon</td>
<td>GFFT-based</td>
<td>iterative</td>
</tr>
<tr>
<td>GRM</td>
<td>generalized</td>
<td>iterative</td>
</tr>
</tbody>
</table>

Table 1. Comparison of Fourier and generalized convolution techniques.

Instead of applying a Fourier matrix to encode the message, we have applied a Pascal matrix and extended the convolution theorem over finite fields. In doing so, we have observed that this formulation leads to the well-known majority logic decoding algorithm. Additional investigations have also considered codes in the context of the wavelet transform (Sakk & Wicker (2003)). The block codes addressed in this chapter have been shown to lend themselves to graph-based iterative decoding strategies (see Table 1). The results derived above suggest that the generalized convolution approach is useful for understanding the systematic introduction of redundancy for the sake of error control.
8. References


The book focuses on Fourier transform applications in electromagnetic field and microwave, medical applications, error control coding, methods for option pricing, and Helbert transform application. It is hoped that this book will provide the background, reference and incentive to encourage further research and results in these fields as well as provide tools for practical applications. It provides an applications-oriented analysis written primarily for electrical engineers, control engineers, signal processing engineers, medical researchers, and the academic researchers. In addition the graduate students will also find it useful as a reference for their research activities.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:
